A Self-paced Learning Algorithm for Change Detection in Synthetic Aperture Radar Images

Ronghua Shang, *Member, IEEE*, Yijing Yuan, Licheng Jiao, *senior Member, IEEE*, Yang Meng and Amir Masoud Ghalamzan, *Member, IEEE*

***Abstract*—Detecting changed regions between two given synthetic aperture radar images is very important to monitor change of landscapes, change of ecosystem and so on. This can be formulated as a classification problem and addressed by learning a classifier, traditional machine learning classification methods very easily stick to local optima which can be caused by noises of data. Hence, we propose an unsupervised algorithm aiming at constructing a classifier based on self-paced learning. Self-paced learning is a recently developed supervised learning approach and has been proven to be capable to overcome effectively this shortcoming. After applying a pre-classification to the difference image, we uniformly select samples using the initial result. Then, self-paced learning is utilized to train a classifier. Finally, a filter is used based on spatial contextual information to further smooth the classification result. In order to demonstrate the efficiency of the proposed algorithm, we apply our proposed algorithm on five real synthetic aperture radar images datasets. The results obtained by our algorithm are compared with five other state-of–the-art algorithms, which demonstrates that our algorithm outperforms those state-of-the-art algorithms in terms of accuracy and robustness.**

***Index Terms*—Change detection, synthetic aperture radar (SAR), self-paced learning.**

I. INTRODUCTION

IMAGE change detection is a technology to detect changed and unchanged regions between images taken from the same place at different times, which helps following studies and analyses [1]. In many civil or military applications such as medical detection and treatment [2, 3], remote sensing [4], and video surveillance [5, 6], image change detection plays a vital role [7, 8]. Change detection in SAR images is getting increased attention in recent years for the imaging characteristics of SAR, such as all-time, all-weather, and large-area [9]. SAR images can provide more information than ordinary optical ones [10], but it suffers from speckle noise [11]. Processing of SAR images with multiplicative speckle noises is very challenging [12]. Most developed unsupervised algorithms used for detecting the changes in SAR images can be divided into three steps [13]. First, a geometric correction and registration are usually implemented to transform two images into a same coordinate system. Next, a difference image (DI) is generated based on two SAR images. Log-ratio operator is a widely used method [14] for this purpose, which transforms a difference image into a logarithmic scale one and converts multiplicative noises into additive ones. Finally, they analyze the DI aiming at forming a classification to classify the changed regions between two SAR images. The quality of the generated DI plays a decisive role in the final result of the change detection of SAR images.

Two methods have been widely used for analyzing DI: (i) clustering methods and (ii) threshold methods. As for the clustering methods, fuzzy c-means clustering (FCM) is one of the most famous and classical one [15]. This method can retain more information than hard clustering. In this regard, Ahmed *et al*. [16] introduced the spatial neighborhood information by modifying the objective function in FCM_S causing a large time complexity. In order to speed up the running time, Szilagyi *et al*. [17] proposed the Enhanced FCM (EnFCM). Krindis Chatzis *et al*. [18] proposed robust fuzzy local information C-means clustering method (FLICM). Its main contribution was using a novel fuzzy local similarity measurement to alleviate the influence of noise and preserve more image

This work was partially supported by the National Natural Science Foundation of China, under Grants 61371201, the National Basic Research Program (973 Program) of China under Grant 2013CB329402, the Program for Cheung Kong Scholars and Innovative Research Team in University under Grant IRT_15R53.

R. Shang, Y. Yuan, L. Jiao, and Y. Meng are with the Key Laboratory of Intelligent Perception and Image Understanding, Ministry of Education, Xidian University, Xi'an 710071, China. (e-mail: rhshang@mail.xidian.edu.cn; 734093492@qq.com). Amir Masoud Ghalamzan is with the Extreme Robotics Lab, University of Birmingham, UK.

detail with no parameter setting. Gong *et al*. [19] proposed a reformulated FLICM (RFLICM) to further reduce the effect of speckle noises by adding a fuzzy factor to the objective function. The clustering methods mentioned above can achieved good change detection maps, but they are affected by the speckle noises [20]. In addition, it is difficult to achieve the balance between preserving the details and suppressing the noise's effect.

Classical threshold methods mainly build a statistical model first for DI, and use algorithms such as Kittler Illingworth (KI) [21] or expectation maximization (EM) algorithm [22] to acquire an appropriate threshold. In this regard, Bruzzone *et al.* [23] proposed to use Gaussian distribution to model DI where an EM algorithm was used to find the threshold. The Markov random field (MRF) [24], the Fisher distribution [25], and the Nakagami distribution [26] were also used to model DI. Furthermore, the multinomial latent model [27] was applied on SAR images. Threshold method is easy to understand and simple to implement, but has a low accuracy. DI is complex because of the speckle noises, so methods mentioned above are hard to establish accurate models for DI.

Li *et al*. [28] recently proposed a new unsupervised algorithm utilizing some known change detection maps and the matching pursuit to learn a dictionary. Gong *et al*. [29] used deep learning to achieve change detection for SAR images. They select samples based on a pre-classification without using difference image. Deep learning was then used to learn high-order features and classify the SAR images. Deep learning has shown promising performance in classification problems and it achieves accurate results. The learning algorithms used to tackle change detection issues in SAR images can avoid the shortcomings of traditional clustering and threshold methods [29] to some extent. In recent years, machine learning methods plays more and more important role in many areas. Such as for handling a quadratic formulation with a pair of equality constraints, an interesting accurate on-line algorithm for training ν-support vector classification was proposed [30]. Two finite mixture models was proposed to capture the structural information of the data from binary classification and obtained good results were obtained [31]. A robust regularization path algorithm for ν-support vector classification was proposed and the proposed algorithm found effective experimental results [32]. Kernel technique was introduced to improve the existing quaternion principal component analysis and the improved algorithm obtained effective results [33]. Nonetheless, classical machine learning methods are sensitive to noises of the samples, and they also easily get stuck into local optima. Although deep learning methods are superior to tradition methods, they are also affected by noises and have a large complexity.

Self-paced learning (SPL) [34, 35] has attracted huge attentions in recent years. SPL is useful for many problems such as specific-class segmentation [36], long-term tracking [37], and visual category discovery [38]. Meng *et al*. [39] gave a theory analysis for SPL, and they proved that SPL is robust to noisy samples and can address local optima problem. Because of the superiority of SPL on classification problems, many variations of SPL have been proposed [40]. Jiang *et al*. [40] proposed SPLD incorporating diversity of samples into basic SPL. Jiang *et al*. [41] used self-paced reranking method to deal with multimedia search problems. They proposed SPCL [42] combining self-paced learning and curriculum learning. In addition, Li *et al*. proposed [43] MLSPL incorporating the self-paced learning strategy into multi-label learning regimes to improve classification accuracy. Li *et al.* [44] proposed SPMTL incorporating self-paced learning into multi-task learning paradigm.

Self-paced learning has excellent performance on the classification problems due to its special learning mechanism. This can overcome the disadvantages of traditional learning algorithms mentioned above. But SAR image suffers from speckle noises and has spatial continuity, and these special characteristics make change detection of SAR image distinguish from other classification problems. Therefore, we propose a new self-paced learning algorithm combined with characteristics of SAR images to deal

with the change detection issues in SAR images in order to achieve more accurate and more robust results. Our algorithm has the following characteristics: (i) this algorithm does not need labeled data. Self-paced learning is a supervised learning method, which needs labeled samples. However, for SAR image change detection, manually labeling each pixel consumes a lot of time and manpower. So the proposed algorithm first uses basic FCM to pre-classify the DI and selects some samples based on the classification result of FCM; (ii) In SAR image change detection, the gap between the number of changed pixels and the number of unchanged pixels is often very large. Our algorithm uniformly selects samples to avoid lacking samples in one specific class. In addition, our algorithm uses spatial continuity of SAR image to ensure the accuracy and diversity of samples; (iii) the feature of selected sample in the proposed algorithm is not a single pixel but a neighborhood, so the learning process can incorporate spatial information to enhance the robustness of the algorithm; (iv) after the classification of the classifier obtained by self-paced learning, a simple filter is used to smooth the final result. So, our proposed algorithm benefits from the characteristics of SAR images and the superiority of self-paced learning.

The rest of this paper is as follows: Section II describes the detail of self-paced learning algorithm for SAR images change detection. Introductions of datasets, evaluation criteria and parameter analysis are presented in section III. Section IV shows the experimental results of the proposed algorithm and five compared algorithms. And the conclusion of this paper is drawn in section V.

II. METHODOLOGY

Consider two co-registered SAR images $I_1=\{I_1(i, j), 1\le i\le A, 1\le j\le B\}$, $I_2=\{I_2(i, j), 1\le i\le A, 1\le j\le B\}$, which are captured from the same place at times t_1 and t_2 respectively. The main purpose of change detection is to identify the change of every pixel at time t_1 and t_2 represented by I_1 and I_2, which actually is a classification problem. This ultimately forms a binary image $I=\{I(i, j), 1\le i\le A, 1\le j\le B\}$ where the size of I, I_1 and I_2 are $A\times B$, $I(i, j)$ is 0 or 1 means that the corresponding pixel is unchanged or changed.

Recently, Bengio *et al*. [45] proposed curriculum learning raising widespread concern in machine learning and computer vision fields. Kumar *et al*. [31] proposed Self-paced learning that can be considered as a subset of the curriculum learning [30]. Inspired from the learning process of human, this approach initially learns from easier samples. Then, it gradually utilizes more complex samples. Many experiments have demonstrated that self-paced learning avoids sticking to a local optimum and results in an effective solution [46]. In this paper, we propose to use self-paced learning combined with the characteristics of SAR images to achieve the change detection of SAR images.

The main task of SPL is to obtain a classifier by minimizing an objective function as follows:

$$\min_{w,v} E(w,v;\lambda)=\sum_{i=1}^{m} v_i L(y_i, f(x_i, w))-\lambda\sum_{i=1}^{m} v_i \qquad (1)$$

$$\text{s.t.}\quad v\in[0,1]^m$$

where m is the number of samples, x_i is the i_{th} sample in the training set, y_i is the label of x_i, $f(x_i,w)$ denotes the model of the classifier and w is the parameter of the classifier. $L(y_i, f(x_i,w))$ is the cost function of x_i and indicates a difference between y_i and $f(x_i,w)$. $\boldsymbol{v}$ is an m-dimension vector, and v_i is the i_{th} element which denotes the difficulty of sample x_i where $v_i=1$ means x_i is "easy", and $v_i=0$ means x_i is "complex". λ is an "age" parameter to determine whether a sample is "easy".

SPL is type of supervised learning which needs labeled samples, but labeling SAR images is a very difficult task in SAR image processing. To avoid this, we propose a unsupervised algorithm to realize change detection for SAR images based on self-paced learning, including the following steps:

1. Adopt log-ratio operator to generate difference image (DI).

2. Use fuzzy c-means clustering method (FCM) to classify DI and obtain an initial result map.

3. Obtain a high-quality training set based on the initial result obtained by FCM.

4.Classify the DI using a trained classifier based on self-paced learning.

5. Smooth the classification result by considering the spatial neighborhood information.

FCM is used to conduct a pre-classification over DI generated by log-ratio operator. Selection of the samples, training of the classifier using self-paced learning and filtering the results are performed as shown in section II.A, II.B, and II.C, respectively. A schematic flowchart of the proposed algorithm is reported in Fig. 1.

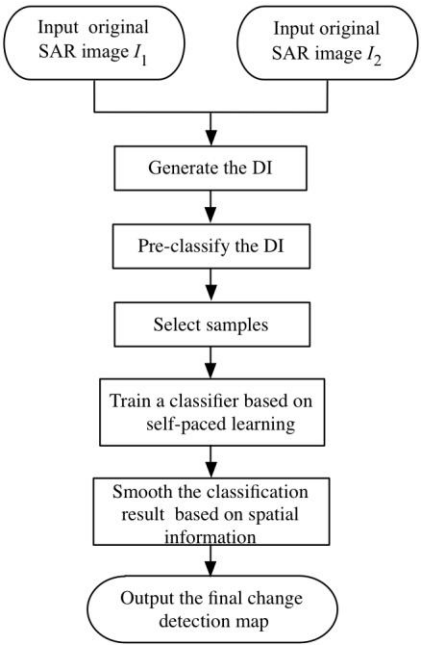


Fig. 1. Framework of the proposed algorithm

A. Selection of The Samples

As FCM just processes single pixel without considering any spatial information for DI's classification, the result is seriously affected by speckle noises. Hence, we need select some "good" pixels to train a classifier using the results obtained by pre-classification of FCM. Training the classifier using high-quality samples produces a very good result, where high quality means that samples in training set both possess accuracy and diversity [29]. Here, we first select a candidate training set including pixels satisfying the condition: they have the label same as most pixels in their neighborhood shown in Fig. 2.

Each circle in Fig. 2 represents the pixel belonging to the changed or unchanged class and each cross shows the pixel belonging to the other class according to the classification performed by FCM. Fig. 2(a), (b) and (c) show the label of the center pixel is similar to the ones of most pixels in its neighborhood based on the pre-classification result. Hence, the center pixel is not considered to be a noise according to the spatial continuity of SAR images.

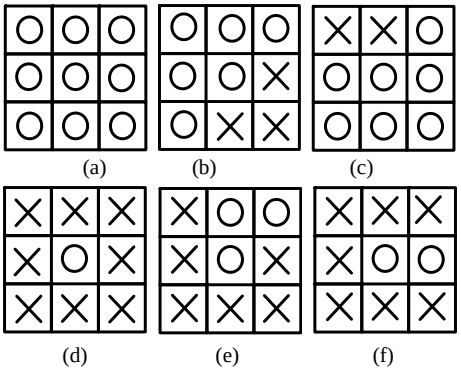


Fig. 2 different situations of a 3×3 neighborhood of pixel *i* after pre-classification; (a), (b) and (c) show pixel *i* is selected as a candidate sample; (d), (e) and (f) show pixel *i* is dropped.

Fig. 2 (d), (e), and (f) show that label of the center pixel is different from the ones of most pixels in its neighborhood. If the label of a pixel and its neighbors are dissimilar, the label of this pixel is considered with high probability to be noise and will be discarded. We adopt a standard to select samples to form a candidate training set, as follows:

$$\frac{t}{s \times s} \geq \alpha \tag{2}$$

where s represents the size of neighborhood centered by one certain pixel, α is a selecting threshold and t is the number of pixels in the neighborhood whose labels are identical to the label of the center pixel. A pixel is considered to be in our candidate training set Tc if the condition in eq. (2) is satisfied. According to the spatial continuity of SAR images [16], selecting "good" samples through this standard is rational. In order to achieve a desired result, we need the value of α to be neither very large nor very small. If α is very small, pixels shown in Fig. 2 (d) may be selected for the training set. Thus, noisy pixel may be selected as a sample. If α is very large, only pixels with neighborhood similar to what is shown in Fig. 2 (a) can be selected. Nonetheless, pixels in edge regions will have small possibility to be selected as a sample, which will cause the final training set loss diversity.

The candidate training set Tc consists of pixels satisfying the condition presented in eq. (2), but the

number of these pixels is large, so we need to select a certain number of pixels from candidate training set to generate the training set. In some SAR images change detection tasks, the number of pixels in changed class and the number of pixels in unchanged class have a big gap, and this gap still exists in the candidate training set *Tc*. If a random selection is carried out, the gap will be easily presented in the final training set. Because pixels in the class with fewer pixels in *Tc* have low probability to be selected, in final training set, the number of samples in this class will be too small. To avoid this, we introduce a uniform-selecting strategy, specifically, we copy pixels in the class with fewer pixels in *Tc* and add these copied pixels into *Tc* to make two class have similar amounts of pixels, then we randomly select a certain number of pixels into the training set *T* to generate final samples. By doing the copy, the number of pixels of the class with fewer pixels in *Tc* will increase, so the probabilities of these pixels to be selected as final samples become larger and the gap between samples' amounts in two classes will be alleviated. The processes of randomly selecting and selecting with uniform-selecting strategy are shown in Fig. 3 (a) and (b), respectively. In this figure, circles represent unchanged pixels while stars represent changed ones, blue stars represent the copied changed pixels. .

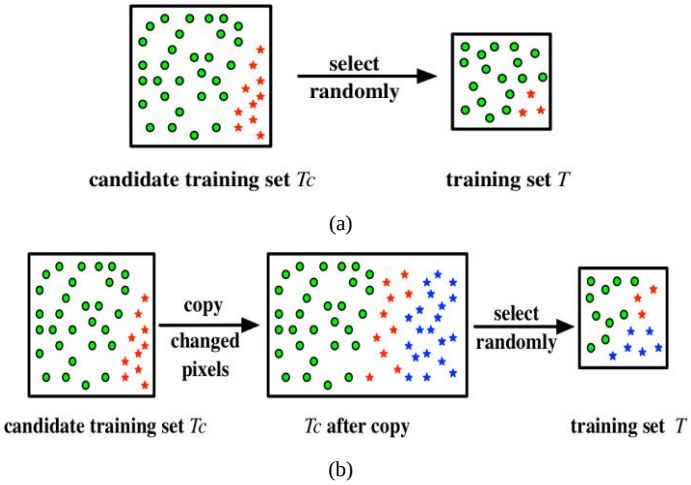


Fig. 3. Samples' selection: (a) The process of randomly selecting; (b) The process of selecting with a uniform-selecting strategy.

As shown in Fig. 3 (a), amount of unchanged pixels is much larger than the amount of changed pixels in candidate training set *Tc*. After selecting randomly, changed samples are few in the training set *T*, so the diversity of samples in this class is hard to meet. Nonetheless, we can avoid such a situation after implementing a copy. Finally, a high-quality training set with a balanced number of samples of each class is obtained, as shown in Fig. 3 (b). After the copy, the number of changed pixels in *Tc* increases, so changed pixels have higher probabilities to be selected. And the in training set *T*, we can see from Fig. 3 (b) that the gap is alleviated a lot. We use a real SAR image dataset, which is shown in Fig. 4 and is called Bern dataset, to validate that diverse samples can be selected by the strategy mentioned above.

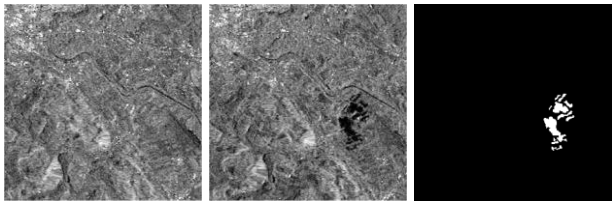


Fig. 4. Bern dataset: (a) and (b) two SAR images captured at two different times; (c) the ground truth map.

Fig. 4 shows two SAR images and the corresponding ground truth map. As shown in Fig. 4(c), most pixels of the two SAR images are unchanged. The results of final samples selected randomly and selected with uniform-selecting strategy on this dataset are reported in Table I.

TABLE I
Results of selected samples

Method	Number of unchanged pixels	Number of changed pixels	Proportion
Ground Truth	89446	1155	77.44
Random Select	9029	31	291.25
Uniform Select	4401	360	12.23

We can see that in the ground truth map, the proportion of the number of unchanged pixels to the number of changed pixels is 77.44, as reported in Table I. Random selection even worsen this and makes the proportion 291.25 where we have only 31 samples belonging to the changed class. On the other hand, using the uniform-selecting strategy helps us to reduce the proportion to 12.23, i.e. samples in unchanged class decreased whereas the number of samples in the other class increased very much.

According to the result reported in Table I, we can get the desired number of samples for each class by uniformly selecting the samples. Besides, by using the standard in eq. (2), most samples have

correct labels. In the final training set of Bern dataset, only 4 out of 4743 samples are wrongly labeled resulting in the accuracy of 99.92%. This demonstrates that the selected samples by our algorithm have high quality. Once a pixel is selected to be a sample, its immediate $a \times a$ neighborhood in *DI* is used to represent this sample, by which the training of the classifier can be provided more information.

B. Classifier Training Based on Self-paced Learning

After obtaining the labeled samples, we can train the classifier based on these samples and SPL. In SPL, the samples utilized to train the classifier are form "easy" to "complex". The criterion to determine whether a sample is "easy" or "complex" is based on a cost function $L(y_i, f(x_i,w))$ in eq. (1) which represents an error between real label and predicted probability by the present hypothesized model and the "age" parameter λ. If the cost value computed for a sample is larger than λ, the existing hypothesis, i.e. the model at this iteration, cannot explain this sample well. Hence, this sample is considered to be "complex" and is not considered for training. Otherwise, the sample is regarded as "easy" and added into the training. At every iteration, λ is increased, and the value of the parameter w of classifier computed at each iteration is used as initial value in the next iteration. λ is similar to the age of human in human learning, hence, we call it "age" [40]. So the iterative process learns from "easy" samples in the initial iterations and it incrementally uses more "complex" samples.

In fact, eq. (1) is a non-convex optimization problem, which may be impossible to directly solve. Alternative convex search (ACS) [47] is often used to solve eq. (1). In detail, ACS is an iterative optimization method where the iteration in ACS is called self-paced iteration. Each self-paced iteration in ACS can be divided into the following two steps:

1) Optimize parameter $\boldsymbol{v} = [v_1, v_2, \ldots, v_m]$

Based on the classifier's parameter w obtained in last iteration, determine v_i as follows:

$$v_i = \begin{cases} 1, & \text{if } L(y_i, f(x_i,w)) < \lambda \\ 0, & \text{otherwise} \end{cases} \qquad i=1, 2, \ldots, m \quad (3)$$

2) Optimize parameter w

After calculating parameter $\boldsymbol{v}$ based on eq. (3), we update parameter w using eq. (4):

$$w = \arg\min_{w} = \sum_{i=1}^{m} v_i L(y_i, f(x_i,w)) \qquad (4)$$

λ is increased in each self-paced iteration by $\lambda=\lambda\times\beta$, where $\beta>1$. The value of λ in the present iteration is larger than the one in last iteration. This allows some more "complex" samples to be added in the present iteration. Parameter w used in the first step uses the parameter w optimized at the previous iteration through the second step. Eventually, we obtain a good classifier by iterating over two steps. The cost function $L(y_i, f(x_i,w))$ need to be convex, hence, we consider the log likelihood cost function used in logistic regression [29]. We assume the samples in our dataset are independent and identically distributed (i.i.d). Hence,

$$\begin{aligned} &L(y_i, f(x_i,w)) = \\ &\quad -(y_i \log(g(w^T x_i)) + (1-y_i)\log(1-g(w^T x_i))) \end{aligned} \qquad (5)$$

where x_i is i_{th} sample with n+1 features, $x_i=(1, x_{i1}, x_{i1}, \ldots, x_{in})^T$, y_i is the label of x_i whose value is 0 (unchanged) or 1 (changed), w is the classifier's parameter, $w=(w_0, w_1, w_2, \ldots, w_n)$. $g(\bullet)$is sigmoid function $g(z)=\frac{1}{1+e^{-z}}$. Logistic regression has been successfully applied to binary classification problems in many domains [48] by using maximum likelihood estimation to solve parameter w as follows:

$$w^* = \arg\min_{w} \sum_{i=1}^{m} L(y_i, f(x_i,w)) \qquad (6)$$

Although the optimization process in logistic regression uses all the samples, in SPL we only use "easy" samples, as eq. (4). Increasing number of samples is considered at the iteration with increased value of λ, and all the samples are considered to be "easy" in the last iteration, i.e. the optimization in eq. (4) will become the same as the one in eq. (6).

To find an optimal solution to the optimization problem in eq. (4) we utilize basic gradient descent [49]. This method easily sticks to local optimum, and the situation is more serious when problem becomes more complex. Besides, the obtained solution by

gradient descent highly depends on the initial values of the parameters. But SPL can avoids local minima [46] which will be demonstrated in section IV. And parameter w is updated based on gradient descent as follows:

$$w_j^{t+1} = w_j^t - \gamma \frac{\partial(\sum_{i=1}^{m} v_i L(y_i, f(x_i, w)))}{\partial w_j} = w_j^t - \gamma(\sum_{i=1}^{m} v_i \frac{\partial(L(y_i, f(x_i, w)))}{\partial w_j}) \quad (7)$$

where t represents current iteration of gradient descent, γ is a step parameter to control the gradient descent, the derivation of eq. (5) is given as follows:

$$\frac{\partial(L(y_i, f(x_i, w)))}{\partial w_j} = ((1-y_i)\frac{1}{1-g(w^T x_i)} - y_i \frac{1}{g(w^T x_i)})\frac{dg(w^T x_i)}{dw^T x} x_i^j = (g(w^T x_i) - y_i)x_i^j \quad (8)$$

where the derivation of sigmoid function $g(w^T x_i)$ is:

$$\frac{dg(w^T x_i)}{dw^T x} = g(w^T x_i)(1 - g(w^T x_i)) \quad (9)$$

A pseudo algorithm of the training of the classifier is reported in Table II.

TABLE II
TRAINING PROCESS OF CLASSIFIER

Algorithm: procedure of the classifier training based on self-paced learning

Input: training set T, max number of self-paced iterations *Maxiter*, max number of gradient descent iterations *Maxgrad*, initial λ, parameter β.

Output: final classifier parameter w^*.

Begin

1. Initialize parameter w randomly, set the self-paced iteration counter to 1;
2. Optimize parameter $\boldsymbol{v} = [v_1, v_2, \ldots, v_m]$ by (3);
3. Optimize parameter w by (7) and (8) through *Maxgrad* gradient descent iterations;
4. Check the current self-paced iteration counter, if it reaches *Maxiter*, w^* equals to current w; otherwise, $\lambda = \lambda \times \beta$, continue to execute step 2;
5. Output the final parameter w^*;

End

The final classifier parameters w^* are obtained after converging to a solution, and eq.(10) is used to calculate the probability that every non-sample pixel in DI belongs to the changed class and the labels of these pixels is determined using eq. (11).

$$P_j = P(y_{zj} = 1 | z_j, w^*) = g((w^*)^T z_j) \quad (10)$$

$$y_{zj} = \begin{cases} 1, & if \quad P_j > 0 \\ 0, & otherwise \end{cases} \quad (11)$$

where z_j represents the j_{th} non-sample pixel in DI, and y_{zj} is the label of z_j predicted by the obtained classifier.

C. Smooth the Classification Result

SAR images are spatially continuous, i.e. each pixel is related to its neighborhood. Nonetheless, we did not consider this in the selection of the samples and training of the classifier. Here, we introduce the local spatial neighborhood information into a filter to smooth the classification result and to further suppress the effect of noises.

After gaining a classification result by the classifier, the classification information of a pixel's neighborhood is used to set the final label of that pixel. The final change-detection label y_i of pixel i is determined by its neighborhood, and can be obtained by eq. (12).

$$y_i = \begin{cases} 1, & if \quad n_{i1} > n_{i2} \\ 0, & otherwise \end{cases} \quad (12)$$

where n_{i1} is the number of pixels whose labels are 1 according to the classifier in neighborhood S_i centered by pixel i, while n_{i2} represents the number of the pixels whose labels are 0. If $n_{i1} > n_{i2}$, most pixels' labels are 1 in S_i, and we set the label of pixel i to be 1 and 0 otherwise. In this way, we can get a smooth change detection map. This helps us to discard the effect of noises on the final result.

Selecting the size of S_i, denoted by r, is very important. If r is large, the algorithm will have high robustness to the noise but pixels in edge regions are likely to be wrongly classified. If r is small, majority of the details of the image is preserved. However, the algorithm is more sensitive to noise resulting in error in labeling. In addition, the edges in the final map are rough. Consequently, regardless of the value of r, which is very large or very small, the accuracy of result decreases. So, we will set r to be 3×3 in the proposed algorithm in order to achieve a balance between preserving details and suppressing noises.

III. EXPERIMENTS SETTING

We use five real SAR images datasets and 5 compared algorithms to test our proposed algorithm. The datasets and evaluation criteria are presented in

section III. A and B. Section III. C illustrates the influence of parameters involved in the proposed algorithm.

A. Datasets and Compared Algorithms

We use 5 other algorithms as compared algorithms to validate the effectiveness of the proposed algorithm, which are logistic regression (LR) [48], FCM [15], MRFFCM [50], FLICM [18], and D_JFCM [29]. D_JFCM is an algorithm that uses JFCM and the restricted Boltzmann machine (RBM) to achieve a promising result without generating a DI. Parameters of MRFFCM and D_JFCM are set referring to the original papers, and there is no parameter setting in FLICM. Besides, five datasets are used for experiments, as shown in Fig. 4 and Fig. 5.

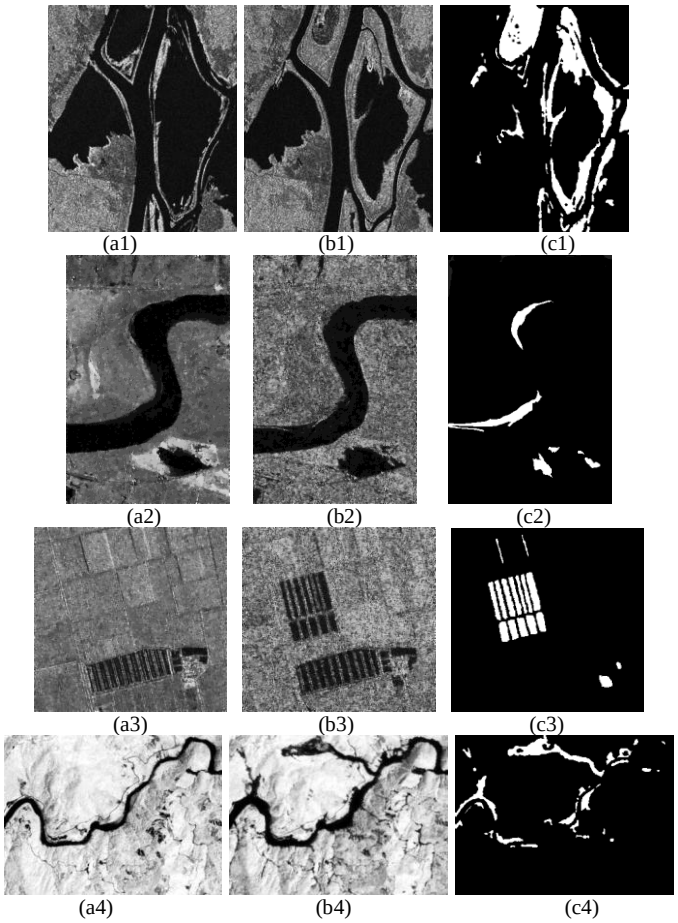


Fig. 5. Datasets. (a1)-(c1) Ottawa dataset. (a2)-(c2) Inland River dataset. (a3)-(c3) Farmland dataset. (a4)-(c4) Shimen Reservoir dataset.

The details of each dataset include the size, the imaging location, imaging times, and the reason why changes happened are listed in Table III.

TABLE III
DETAILS OF DATASETS

Name	size	Location	Time t_1	Time t_2	Change reason
Bern	301×301	Bern, Switzerland	April, 1999	May, 1999	flood
Ottawa	290×350	Ottawa, Canada	May, 1997	August, 1997	rain
Inland River	444×291	Yellow River estuary	June 2008	June, 2009	farming
Farmland	291×306	Yellow River estuary	June 2008	June, 2009	farming
Shimen Reservoir	252×349	Taiwan, China	August, 2004	September, 2004	typhoon

B. Evaluation Criteria

In this paper, we use both qualitative and quantitative analysis to evaluate the change detection results on all datasets. Qualitative analysis means that we compare the maps obtained by all algorithms with the ground truth maps by human eyes, which is not very accurate.

We adopt the standard proposed in [51] for the quantitative analysis. First, using the ground truth map we count all the pixels, denoted by *N*, the changed pixels, denoted by *Nc*, and the unchanged pixels, denoted by *Nu*. Then, we compare the obtained binary image to the ground truth map pixel by pixel and compute the number of unchanged pixels which are undetected, denoted by *FP*, and the number of changed pixels which are undetected, denoted by *FN*. In addition, the numbers of changed pixels and unchanged pixels that are correctly labeled, are denoted by *TP* and *TN*, respectively. These can be computed using eq. (13).

$$\begin{cases} TP = Nc - FN \\ TN = Nu - FP \end{cases} \tag{13}$$

FP, *FN*, *TP*, and *TN* are not enough to evaluate more precisely the change detection results. So we compute the percentage of correct classification (*PCC*) by eq. (14).

$$PCC = \frac{TP + TN}{N} \tag{14}$$

PCC represents the proportion of pixels classified correctly to total pixels. In general, we expect to have a large *PCC* if the algorithm can effectively classify the dataset. We compute the overall error (*OE*) by (15).

$$OE = \frac{FP + FN}{N} \tag{15}$$

OE is the proportion of pixels classified wrongly

to total pixels. If an algorithm is effective, it should have a low *OE*. It is worth noting that the sum of *PCC* and *OE* is equal to 1.

However, standard widely used is the kappa coefficient (*KC*), which is an evaluation criterion in the image segmentation [52] computed as follows:

$$KC = \frac{PCC - PRE}{1 - PRE} \tag{16}$$

where

$$PRE = \frac{(TP + FP) \times Nc + (FN + TN) \times Nu}{N \times N} \tag{17}$$

If two algorithms have similar *PCC* and *OE*, their *KC* may differ significantly because the calculation of *KC* considers more detail information. *KC* is a number between 0 and 1. The larger the value of *KC*, the better the result of change detection.

C. Parameter Analysis

1) Threshold α

After pre-classifying DI by FCM, some samples are selected for training the classifier. The standard used in the selection of the samples is related to the threshold α. This threshold is essential as it determines the quality of samples. If parameter α is very large, most of the samples will be pixels in homogeneous regions according to eq. (2) where the pixels in edge regions are not included. In contrast, if α is too small, there may be many noises in the training set and the necessary accuracy of the samples will not be satisfied. We consider α being 0.05, 0.10, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8, and 0.85, and use five datasets mentioned above to test the influence of threshold α on the kappa coefficient where the corresponding result is shown in Fig. 6.

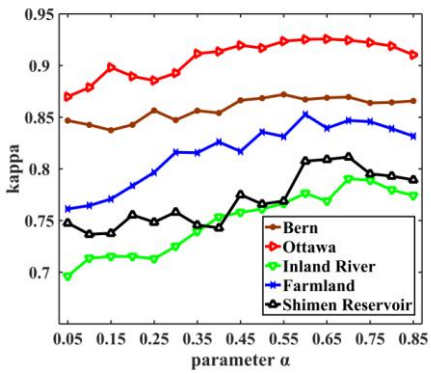


Fig. 6. α versus kappa

As we can see in Fig. 6, when α increases from 0.05 to 0.85, the kappa coefficients on all datasets by the proposed algorithm increase first, and then decrease slowly. When α is small, i.e. α<0.45, kappa coefficients are low, the reason is that the selected samples may suffer serious noises and the accuracy of samples cannot be guaranteed, so the classifier has a weak performance. But when α continues increasing, i.e. when α>0.7, kappa coefficients will slightly go down. The reason is that some pixels in edge regions cannot be selected as samples, so the samples under this situation may not meet the diversity. In Fig. 6, kappa coefficients of the Inland River, Farmland, and Shimen Reservoir datasets are influenced by α obviously unlike other two datasets, this is because there exists many changed pixels in edge regions on these three datasets. But the uniform-selecting strategy and spatial neighborhood feature used can ameliorate the diversity of the samples to some extent, so the kappa coefficients are just a little lower with large α.

2) Parameter β

In self-paced learning, complex samples will be increasingly added to learning process iteratively. λ determines whether a sample is "easy" or "complex" where a sample with a larger cost function value than λ is regarded as "complex" and is not included in the training of the classifier. Otherwise, this sample is "easy" and will be included in the training phase. λ is a parameter whose value is changing during self-paced learning iterations. It is quite small in the beginning and gradually increases during the learning process. Parameter β controls the increasing speed of λ as well as the increasing speed of samples involved in the training. If β is very large, the difficulty of samples involved in the training process increases sharply during the iterations; i.e. if a few "easy" samples are used to train the classifier at the present iteration, much more "complex" samples will be added to the training process based on a large λ in the next iteration; If β is very small, the algorithm needs to take a long time in order to generate a good result. In addition, there may be some samples cannot be involved in the training in the last iteration. Therefore, we consider β to be 1.0,

1.2, 1.4, 1.6, 1.8, and 2.0 with initial λ equal to 0.1 to test the influence of parameter β on results of all datasets during 5 self-paced iterations. The changes of the number of samples involved in the training during 5 iterations on five datasets are shown in Fig. 7 for different values of β.

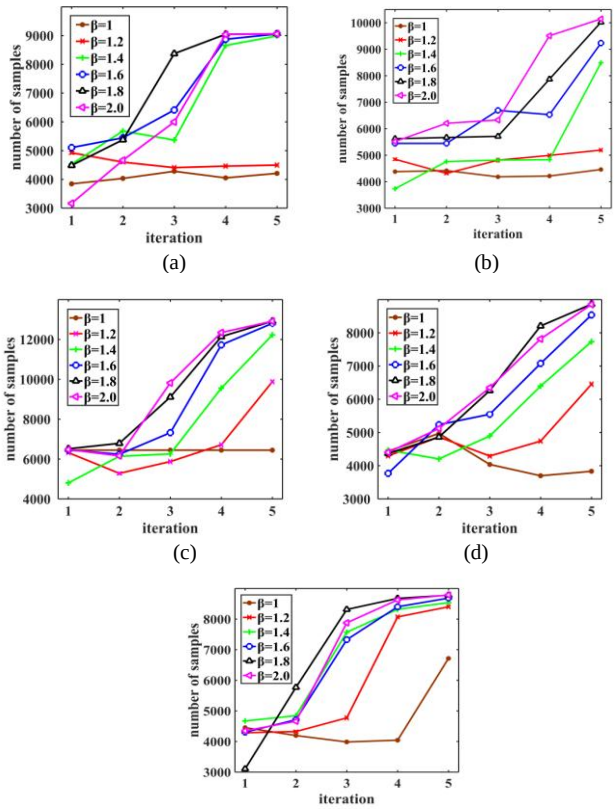


Fig. 7. Number of samples involved in the training at each iteration for different β. (a) On Bern dataset. (b) On Ottawa dataset. (c) On Inland River dataset. (d) On Farmland dataset. (e) On Shimen Reservoir dataset.

During the experiments, we select 10% of all pixels on each dataset to form the training set. In Fig. 7, we can see that for all dataset, the number of samples used to train the classifier increases at different speeds with different β during the iteration. When β is small, such as 1 or 1.2, shown as yellow and green lines in Fig. 7, samples involved in the training increase slow during the iterations. Even in the last iteration, especially on the Bern and Ottawa datasets, the number of samples involved in the training is very few as shown in Figure 11 (a) and (b). In this way, the final obtained classifier cannot achieve good results because of the lack of samples. When β is larger, such as 1.4 and 1.6, shown as the green and blue lines in Fig. 7, samples involved in the training increase in a gentle speed and in the last iteration, all samples can participate in the training. However, when β continues to increase such as 1.8 or 2 as black and pink lines in Fig. 7, λ will change a lot in two consecutive iterations and will have large value in early iterations. As λ is a crucial measurement to judge whether a sample is "easy", large λ will make more "complex" samples involved in the training, but the model in early iterations may not be mature enough to learn from these samples. And Fig.8 shows the kappa coefficients on all datasets with different β after 5 self-paced iterations.

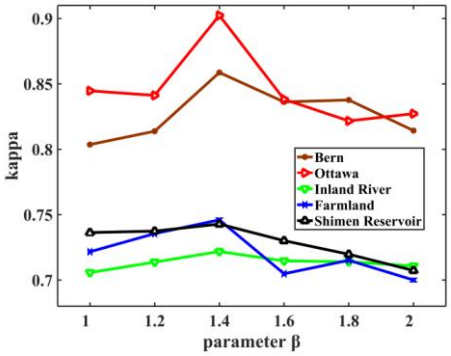


Fig. 8. β versus Kappa coefficient after 5th self-paced iteration.

As shown in Fig. 8, when β is small, kappa coefficients on all data sets are a little low for the lack of samples involved in the training. And kappa coefficients increase with the increase of β, but when β continues to increase, the increase speed of λ and samples involved in the training will be too fast, which results in the kappa coefficients' decrease. The total number of self-paced iterations is 5, different β has great influence on the kappa coefficients in Fig. 8. However, when we increase the number of self-paced iterations to be 10, the kappa coefficients will not be so sensitive to β, as shown in Fig. 9.

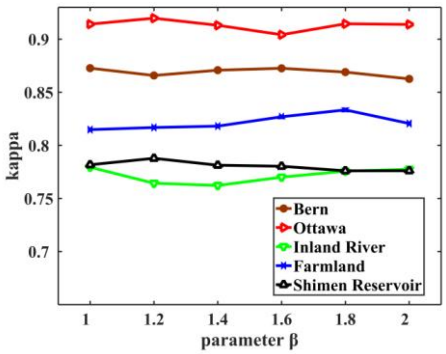


Fig. 9. β versus Kappa coefficient after 10th self-paced iteration

From Fig. 9 we can conclude that if iteration number increases, parameter β will have a slight effect on the final result for all datasets. Compared to Fig. 8, the kappa coefficients are much more stable than that in Fig. 8 with different β. Even though β is small, λ can be large enough after enough iterations so that all samples can be involved in the training. But *KC* will still goes down a little if β is very large, the reason is that under this situation samples involved in the training may be too "complex" to be learned by the current model.

IV. Experimental Study

We performed a series of experiments using the datasets and algorithms mentioned in the previous sections. In this section, we only use 10% of all the pixels in a dataset for training the classifier. The threshold α is 0.7 and features of each sample are gray values of pixels in the selected pixel's 5×5 neighborhood of DI.

We need to set the initial λ and parameter β of self-paced learning before the experiment. The initial λ determines samples involved in the initial self-paced iteration. If λ is large, samples involved in the training may be too "complex" in the initial iteration. On the other hand, if λ is small, the number of samples involved in the training is too small to meet the necessary diversity for our algorithm. In eq. (5), the value of cost function for each sample is from 0 to positive infinity. So, we set λ to be 0.1 for all the datasets. According to the argument in the previous section, we set β to be 1.1 and 15 times self-paced iterations are performed allowing the number of samples involved in the training smoothly increases. This setting is empirically proved to make promising results. We developed all the experiments using MATLAB R2013 on a machine with Intel core I3 2.30-GHz CPU and 4-GB RAM.

We applied our algorithm on five datasets. To validate the obtained results we also use five other algorithms and compare the results in two ways: (i) final binary result maps are displayed where white indicates the changed region and black indicates unchanged region; (ii) the five criteria *FP*, *FN*, *OE*, *PCC* and *KC* are utilized to compare the results of the proposed algorithm and 5 other algorithms. In addition, we provide evidence of the computation cost for all the algorithms.

A. Results on Bern dataset

Fig. 14 shows the change detection maps of Bern dataset using the 5 compared algorithms and our proposed algorithm. As shown in this figure, the difference between the map of LR and the ground truth map is very large. We can infer that the simple LR stuck in a local optimum for training the classifier by using all the samples.

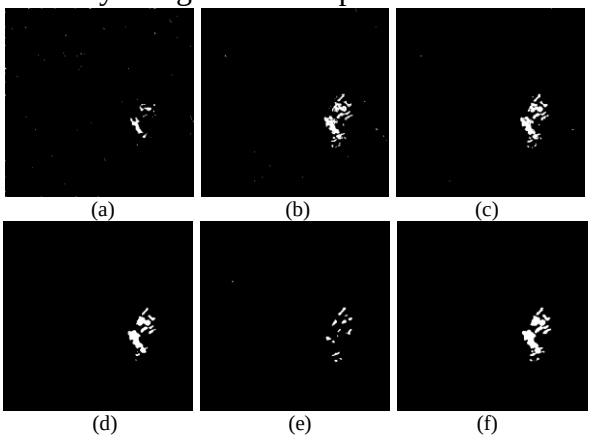


Fig. 10. Change detection maps on Bern dataset by: (a)LR, (b) FCM, (c) MRFFCM, (d) FLICM, (e) D_JFCM, and (f) the proposed algorithm.

Moreover, Fig. 10 (b) shows that the map of FCM contains a lot of noises, and it does not result in a good regional continuity because FCM does not use the spatial information of the image. On the other hand, MRFFCM introduces an improved energy function into FCM to suppress noises and to preserve details. Fig. 10 (c) demonstrates that MRFFCM is good at preserving details, but its map still contains a lot of noises. FLICM use a neighborhood item to suppress the effect of noise. There are few isolated noises in the map as shown in Fig. 10 (d). Nonetheless, FLICM has a limited capability to detect smaller changed areas. For example, it could not detect some discrete white areas at right bottom of the map. Fig. 10 (e) shows that D_JFCM wrongly classifies many pixels as unchanged regions because the gaps of samples' amounts between two classes are not considered in the process of samples' selection. The training process by deep learning lacks samples in changed class, so the final classifier cannot achieve effective detection in changed regions. Fig. 10 (f) shows that the proposed

algorithm is not only effective to suppress the noise, but also very effective to preserve details. In the map, we see that the proposed algorithm detects some small changed areas.

It is not feasible to compare precisely the obtained results using human eyes if the results of different algorithms are very similar. Consequently, we need to compare the results quantitatively as reported in Table IV.

TABLE IV
QUANTITATIVE RESULTS ON BERN DATASET

Criteria / Method	*FN*	*FP*	*OE*%	*PCC*	*KC*	t/s
LR	848	85	1.03	0.9897	0.3930	8.1978
FCM	123	350	0.52	0.9948	0.7703	**5.9028**
MRFFCM	63	281	0.38	0.9962	0.8337	33.0796
FLICM	**36**	293	0.36	0.9964	0.8379	11.8626
D_JFCM	893	**52**	1.04	0.9896	0.3532	98.3740
The proposed algorithm	131	154	**0.31**	**0.9969**	**0.8738**	55.7557

It can be seen from Table IV that the learning process of LR results in a local optimum, so the *KC* of LR is very low. And a lot of changed regions are wrongly classified as shown in Fig. 10 (a) resulting in a high *FN*. Similar to the result of LR, D_JFCM also has a high *FN* because deep learning cannot learn enough features with few samples in changed class. Furthermore, a lot of pixels in changed regions are not detected by D_JFCM, making the *KC* of D_JFCM very low. The performance of FCM is affected by noises and has the largest *FP* shown in Table IV. MRFFCM and FLICM have similar results. They both have high value of *KC*. However, our proposed algorithm results in the best *KC*. In contrast to D_JFCM, our *FN* and *FP* on Bern dataset are very close because our algorithm considers the diversity of samples when selecting them. Regarding the running time, the proposed algorithm is not the fastest one on Bern dataset, but is much swifter than D_JFCM with a better performance.

B. Results on Ottawa dataset

Fig. 11 shows the maps obtained by five algorithms on Ottawa dataset. This figure shows that nearly all maps are close to the ground truth map except LR. Specifically, Fig. 11 (a) shows that the map obtained by LR is very poor because the original LR suffers from noises seriously and it easily sticks in a local optimum. Although FCM preserves many details in the final map, it has poor robustness to noises, as shown in Fig. 11 (b). The map of MRFFCM is shown in Fig. 11 (c), which is very close to the ground truth map. MRFFCM achieves a more accurate detection than other algorithms especially for the changed regions in the upper part of the image. But the resulting map still contains a small amount of noises. The result of FLICM shows that this algorithm efficiently suppresses the noises, but it misclassifies many changed pixels in top part as shown in Fig. 11 (d).

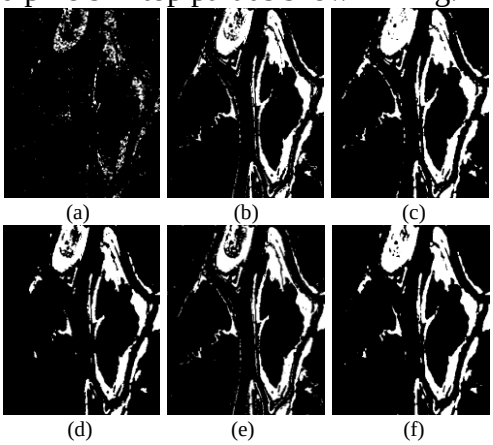


Fig. 11. Change detection maps on Ottawa dataset by: (a)LR, (b) FCM, (c) MRFFCM, (d) FLICM, (e) D_JFCM, and (f) the proposed algorithm.

D_JFCM also misclassifies these pixels. Moreover, the map of D_JFCM contains a lot of noises as shown in Fig. 11 (e). Fig. 11 (f) shows that the proposed algorithm also has pretty good result not only in terms of suppressing the noises but also in terms of preserving details. Quantitative comparison of the results obtained by different algorithms on Ottawa dataset is shown in Table V.

Table V shows that LR has the worst results with a very high FN because the poor classifier obtained by LR wrongly classifies a large number of changed pixels. The *KC* of FCM is higher than that of FLICM because this dataset has many small, limited, and changed pixels in the ground truth map, FLICM has lost a lot of detailed information while suppressing the noises. D_JFCM still has high *FN* because the training of classifier lacks samples of changed class. So the *KC* of D_JFCM is also very low. MRFFCM can achieve precise detections with a high *KC* because it uses MRF, but it is still worse than the proposed algorithm. It can be seen from Table V that the algorithm has very close *FN* and *FP*. Furthermore, it has the lowest *OE* and the highest *KC* because a good classifier is learned by SPL.

TABLE V
QUANTITATIVE RESULTS ON OTTAWA DATASET

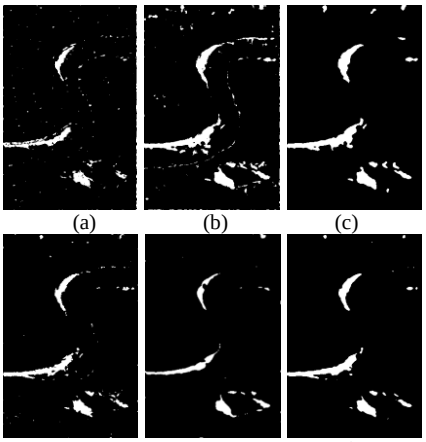
Criteria / Method	*FN*	*FP*	*OE%*	*PCC*	*KC*	t/s
LR	13090	**164**	13.06	0.8694	0.2711	7.8240
FCM	853	2015	2.83	0.9717	0.8907	**3.8312**
MRFFCM	1703	666	2.33	0.9767	0.9146	32.4224
FLICM	**131**	2664	2.75	0.9725	0.8895	6.2948
D_JFCM	3034	1078	4.05	0.9595	0.8399	113.147
The proposed algorithm	894	1010	**1.88**	**0.9812**	**0.9293**	58.7057

D_JFCM has the longest computation time on Ottawa dataset because of the deep learning framework. In addition, MRFFCM has long computation time since it calculates a Markov energy equation at each iteration. Although the proposed algorithm takes longer time to converge than most of the other algorithms, it finally obtains the best *KC* and *OE*.

C. Results on Yellow River Estuary dataset

1) Results on Inland Water dataset

The qualitative results obtained by the compared algorithms and our proposed one on Inland River dataset are displayed in Fig. 12.



(a) (b) (c) (d) (e) (f)

Fig. 12. Change detection maps on Inland River dataset by: (a) LR, (b) FCM, (c) MRFFCM, (d) FLICM, (e) D_JFCM, and(f) the proposed algorithm.

The map obtained by LR is severely affected by noises and is not smooth enough, as shown in Fig. 12 (a). The map of FCM also contains a lot of noises, but it retains the details to a large extent, as shown in Fig. 12 (b). The map obtained by MRFFCM is close to the ground truth map, as shown in Fig. 12 (c), but there is a lot of noises in the top-right of the map. Fig. 12 (d) shows that FLICM achieves a map with many noises. D_JFCM has achieved a good result with a bit of noises. But Fig. 12 (e) shows that D_JFCM misclassifies many changed pixels because the samples are not balanced. Although map of the proposed algorithm contains a small amount of noises shown in Fig. 12 (f), it retains more precise details. The quantitative results of five algorithms and our proposed algorithm are shown in Table VI.

TABLE VI
QUANTITATIVE RESULTS ON INLAND RIVER DATASET

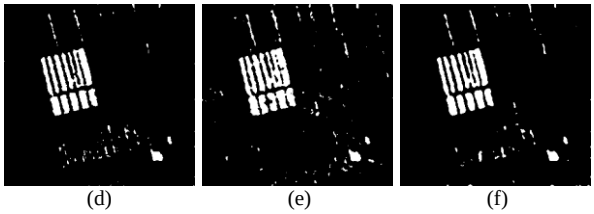
Criteria / Method	*FN*	*FP*	*OE %*	*PCC*	*KC*	t/s
LR	*1690*	*2012*	*2.87*	*0.9713*	*0.5660*	9.9719
FCM	326	3947	3.31	0.9669	0.6320	**2.5330**
MRFFCM	305	2484	2.16	0.9784	0.7283	49.6091
FLICM	793	1381	1.68	0.9832	0.7524	10.0796
D_JFCM	1365	**634**	1.55	0.9845	0.7351	180.2740
The proposed algorithm	**804**	913	**1.33**	**0.9867**	**0.7939**	73.7318

According to the results reported in Table VI, both FCM and LR have poor *KC* because they are affected by noises seriously. MRFFCM also suffers from noises and wrongly classifies pixels in the upper part of the map, which results in a high *FP*. FLICM and D_JFCM have close *KC*, but their *FN* and *FP* are very different because FLICM suffers from noises and misclassifies unchanged pixels. In contrast, D_JFCM misclassifies changed pixels due to the unbalanced samples. Our proposed algorithm achieves the highest *KC* and has close *FN* and *FP*. We can conclude that the computation time of the proposed algorithm is much less than D_JFCM and it achieves the best result in the mean time.

2) Results on Farmland dataset

The binary maps on Farmland dataset obtained by five algorithms are shown in Fig. 13. In particular, LR, FCM, and MRFFCM are not robust to noises and their maps contain a lot of noises, as shown in Fig. 13 (a), (b), and (c). The maps obtained by FCM and MRFFCM are similar. The map in Fig. 13 (d) shows that FLICM can suppress the noises much better than others, but in the middle and lower part of the map, a lot of unchanged pixels are misclassified.

(a) (b) (c)



(d) (e) (f)

Fig. 13. Change detection maps on Farmland dataset by: (a)LR, (b) FCM, (c) MRFFCM, (d) FLICM, (e) D_JFCM, and (f) the proposed algorithm.

The map obtained by D_JFCM is also affected by noises, but it is more capable to distinguish the middle and lower parts in the map, as shown in Fig. 13 (e). The map of the proposed algorithm is similar to the one obtained by D_JFCM but slightly better than that of FLICM, as shown in Fig.13 (f). Table VII reports the quantitative results obtained by all algorithms on Farmland dataset, thus we can further evaluate performance of each algorithm.

TABLE VII
QUANTITATIVE RESULTS ON FARMLAND DATASET

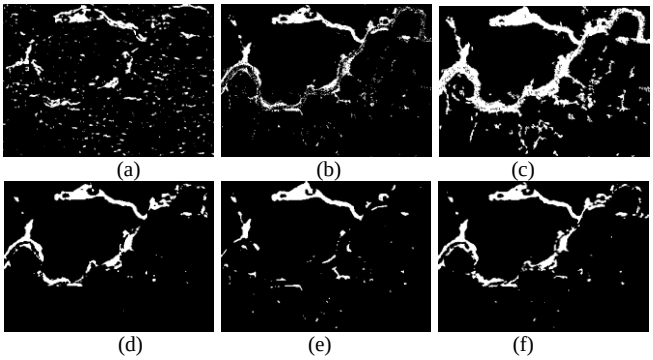
Criteria / Method	*FN*	*FP*	*OE* %	*PCC*	*KC*	t/s
LR	2924	2240	5.80	0.9429	0.4613	8.0795
FCM	583	2390	3.34	0.9666	0.7475	**3.7422**
MRFFCM	**505**	2870	3.79	0.9621	0.7248	34.1391
FLICM	960	**611**	**1.76**	0.9824	0.8410	6.0237
D_JFCM	1296	690	2.23	0.9777	0.8147	148.9340
The proposed algorithm	825	779	1.80	**0.9820**	**0.8419**	54.8372

The quantitative results in Table VII shows that LR gets the worst results, and FCM has close results to MRFFCM with low *FN* and high *FP* because they are both sensitive to noises. In addition, many unchanged pixels are misclassified as shown in Fig. 13 (b) and (c). The quantitative results obtained by D_JFCM are good because the number of samples in different class is relatively balanced in this dataset. Moreover, the gap between *FN* and *FP* values is not very big. The results obtained by FLICM and our proposed algorithms are very close with high *KC*. But our proposed algorithm is a little better than FLICM because *FN* and *FP* are closer in our proposed algorithm.

In terms of computation cost of different algorithms, our proposed algorithm requires more time than other algorithms but D_JFCM because the selection of the samples and the judgment of the difficulty on each sample are computationally expensive. FLICM algorithm can achieve a result similar to the ones by our proposed algorithm in a short time on Farmland dataset, as shown in Table VII. However, our proposed algorithm is more efficient in comparison with D_JFCM.

D. Results on Shimen Reservoir dataset

Five qualitative results obtained by six algorithms on Shimen Reservoir dataset are displayed in Fig. 14. LR falls into a local optimum and achieves a very bad map as shown in Fig. 14 (a). In addition, the map obtained by FCM, shown in Fig. 14 (b), contains a lot of noises because it has poor robustness. In addition, the map obtained by MRFFCM on this dataset is very bad. MRFFCM is also sensitive to noises and misclassifies many unchanged pixels. The map of FLICM, shown in Fig. 14 (d), shows that FLICM is effective to suppress noises but loses many details because there are many slimmer and smaller areas need to be detected in this dataset.



(a) (b) (c)

(d) (e) (f)

Fig. 14. Change detection maps on Shimen Reservoir dataset by: (a)LR, (b) FCM, (c) MRFFCM, (d) FLICM, (e) D_JFCM, and (f) The proposed algorithm.

D_JFCM is not capable of detecting the changed regions with the samples, which are not uniform, as shown in Fig. 14 (e). Our proposed algorithm is not as good as FLICM in suppressing noises, as shown in Fig. 14 (f), but it can retain more details. The quantitative results on Shimen reservoir dataset are reported in Table VIII.

TABLE VIII
QUANTITATIVE RESULTS ON SHIMEN RESERVOIR DATASET

Criteria / Method	*FN*	*FP*	*OE* %	*PCC*	*KC*	t/s
LR	2911	3594	7.40	0.9260	0.3322	7.3136
FCM	683	1474	2.45	0.9755	0.7807	**3.5685**
MRFFCM	**25**	8003	9.13	0.9087	0.5055	27.6005
FLICM	817	1139	2.22	0.9778	0.7923	6.8735
D_JFCM	1748	**485**	2.54	0.9746	0.7212	79.6070
The proposed algorithm	436	1404	**2.09**	**0.9791**	**0.8159**	54.1988

Results reported in Table VIII correspond to the maps in Fig. 14. This table shows that LR has worst

result. In addition, the result of MRFFCM is also very poor with a very high *FP* and a very low *FN*, this algorithm is good at preserving details but sensitive to noises. And quantitative results of FCM and FLICM are close. FLICM is good at suppressing noises but loses much information meanwhile. D_JFCM has a high *FN* due to the diversity of changed samples cannot be satisfied in this dataset. Our proposed algorithm has the highest *KC*, but the *FN* and *FP* have an obvious gap because in this dataset, there are some slim and small changed regions. The training process uses the neighborhood as feature of a sample, so when the neighborhood is large, the classifier is easy to wrongly classify pixels. Likewise the previous four datasets, computation time of the proposed algorithm is shorter than D_JFCM on this dataset with the best *KC*.

E. Time complexity of the proposed algorithm

In this part, we will talk about the time complexity of the proposed algorithm. As illustrated in section II, the proposed algorithm includes DI' generation, initial FCM clustering, samples' selection, the classifier's training, and final smooth. So the time complexity of the proposed algorithm is the sum of time complexities of these different processes. Let N represent the number of the total pixels in ground truth map, n is the number of features of a sample, c=2 is class number of FCM. w_s and w_f are the sizes of neighborhood window and smooth window respectively, and H_f, H_s, and H_g are the numbers of FCM iteration, self-paced iteration, and the gradient descent iteration respectively The time complexity of each process in the proposed algorithm is shown in Table IX.

TABLE IX
COMPLEXITY OF EACH PROCESS

Process	Complexity	Process	Complexity
DI's generation	$O(N)$	FCM clustering	$O(c^2H_fN)$
Samples' selection	$O(w_s^2N)$	Classifier's training	$O(n^2H_sH_gN)$
Smooth	$O(w_f^2N)$	Proposed algorithm	$O(n^2H_sH_gN)$

Considering that the number of a sample's features n is larger than w_s and w_f, so the proposed algorithm has the same time complexity as the classifier's training which is $O(n^2H_sH_gN)$ as shown in Table IX.

V. CONCLUSION

In this paper, we presented a novel algorithm to detect the changes in SAR images using self-paced learning. First, we utilize FCM to pre-classify the difference image (DI). Then, we select some pixels with a uniform-selecting strategy across the DI resulting in a high-quality training set. Next, self-paced learning is used to train a classifier. Finally, local spatial information is adopted to smooth the classification result. We demonstrate the effectiveness of the proposed algorithm by some experiments using five real SAR images datasets. We analyze and study the effect of different parameters used in our proposed algorithm on the final results, namely threshold α and parameter β. Furthermore, to validate the results we compare the results obtained by our algorithm with Logistic Regression, FCM, MRFFCM, FLICM and D_JFCM. In addition, all the results are compared with the ground truth maps showing that our proposed algorithm outperforms other state-of-art algorithms both qualitatively and quantitatively.

ACKNOWLEDGEMENT

We would like to express our sincere appreciation to the editors and the anonymous reviewers for their insightful comments, which have greatly helped us in improving the quality of the paper.

REFRENCES

[1] R. J. Radke, S. Andra, O. Al-Kofahi, and B. Roysam, "Image change detection algorithms: a systematic survey," *IEEE Trans. Image Process*., vol. 14, no. 3, pp. 294–307, Mar. 2005.

[2] Y. Zheng, B. Jeon, D. Xu et. al, "Image segmentation by generalized hierarchical fuzzy C-means algorithm," *Journal of Intelligent and Fuzzy Systems*, vol. 28, no. 2, pp. 961–973, 2015.

[3] J. Li, X. L. Li, B. Yang, and X.M. Sun, "Segmentation-based Image Copy-move Forgery Detection Scheme," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 3, pp. 507–518, Mar. 2015.

[4] B. J. Chen, H. Z. Shu, G. Coatrieux, G. Chen, X. M. Sun, and J. Coatrieux, "Color image analysis by quaternion-type moments," *Journal of Mathematical Imaging and Vision*, vol. 51, no. 1, pp. 124–144, 2015.

[5] R. H. Shang, L. P. Qi, L. C. Jiao, R. Stolkin, et al. Change detection in SAR images by artificial immune multi-objective clustering. *Engineering Applications of Artificial Intelligence*, vol. 31, pp. 53–67, 2014.

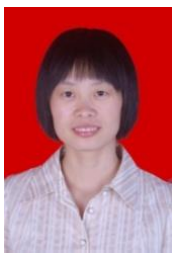
[6] S. S. Ho and H. Wechsler, "A martingale framework for detecting changes in data streams by testing exchangeability," *IEEE Trans. Pattern Anal. Mach. Intell*., vol. 32, no. 12, pp. 2113–2127, Dec. 2010.

[7] R. Shang, P. Tian, L. Jiao, R. Stolkin, et al. A spatial fuzzy clustering algorithm with kernel metric based on immune clone for SAR image segmentation. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 9, no. 4, pp. 1640–1652, 2016.

[8] L. Bruzzone and D. F. Prieto, "An adaptive semiparametric and context-based approach to unsupervised change detection in multitemporal remote-sensing images," *IEEE Trans. Image Process*., vol. 11, no. 4, pp. 452–466, Apr. 2002.

[9] Q. Gao, Y. Lu, D. Sun, Z. Sun, and D. Zhang, "Directionlet-based denoising of SAR images using a Cauchy model," *Signal Process.*, vol. 93, no. 5, pp:1056–1063, 2013.

[10] S. Chitroub, A. Houacine, and B. Sansal, "Statistical characterisation and modelling of SAR images," *Signal Process.*, vol. 93, no. 5, pp:1056–1063, 2013.
[11] S.M. Ali and R.E. Burge, "New automatic techniques for smoothing and segmenting SAR images," *Signal Process.*, vol. 14, no. 4, pp. 335–346, 1988.
[12] E. E. Kuruoglu and J. Zerubia, "Modeling SAR images with a generalization of the Rayleigh distribution," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 527–533, Apr. 2004.
[13] Y. Bzai, L. Bruzzone, and F. Melgani, "An unsupervised approach based on the generalized Gaussian model to automatic change detection in multitemporal SAR images," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 4, pp. 874–887, Apr. 2005.
[14] F. Bovolo and L. Bruzone, "A detail-preserving scale-driven approach to change detection in multitemporal SAR images," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 12, pp. 2963–2972, Dec. 2005.
[15] A. Ghosh, N. S. Mishra, and S Ghosh, "Fuzzy clustering algorithms for unsupervised change detection in remote sensing images," *Inform. Sci.*, vol. 181, no. 4, pp. 699–715, Feb. 2011.
[16] M. N. Ahmed, S. M. Yamany, N. Mohamed, A. A. Farag, and T. Moriarty, "A modified fuzzy c-means algorithm for bias field estimation and segmentation of MRI data," *IEEE Trans. Med. Imag.*, vol. 21, no. 3, pp. 193–199, Mar. 2002.
[17] L. Szilagyi, Z. Benyo, S. M. Szilagyi, and H.S. Adam, "MR brain image segmentation using an enhanced fuzzy c-means algorithm," *IEEE EMBS*, 2003, pp. 724–726.
[18] S. Krinidis and V. Chatzis, "A robust fuzzy local information C- means clustering algorithm," *IEEE Trans. Image Process.*, vol. 19, no. 5, pp. 1328–1337, May. 2010.
[19] M. Gong, Z. Zhou, and J. Ma, "Change detection in synthetic aperture radar images based on image fusion and fuzzy clustering," *IEEE Trans. Image Process.*, vol. 21, no. 4, pp. 2141–2151, Apr. 2012.
[20] J. Feng, L.C. Jiao, X. Zhang, M. Gong, T. Sun, "Robust non-local fuzzy c-means algorithm with edge preservation for SAR image segmentation," *signal process.*, vol. 54, no. 7, pp. 4287–4301, Jul. 2016.
[21] J. Kitler, and J Ilingworth, "Minimum error thresholding," *Pattern Recognit.*, vol. 19, no. 1, pp.41-47, 1986.
[22] A. Dempster, "Maximum likelihood from incomplete data via the EM algorithm," *J. Roy. Statist. Soc.*, vol. 39, no. 1, pp. 1–38, 1977.
[23] L. Bruzzone and D. F. Prieto, "Automatic analysis of the difference image for unsupervised change detection," *IEEE Trans. Geosci. Remote Sens.*, vol. 38, no. 3, pp. 1171–1182, May. 2000.
[24] H. Derin and H. Elliott, "Modeling and segmentation of noisy and textured images using Gibbs random fields," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 9, no. 1, pp. 39–55, Jan. 1987.
[25] C. Tison, J. M. Nicolas, F. Tupin, and H. Maitre, "A new statistical model for Markovian classification of urban areas in high-resolution SAR images," *IEEE Trans. Geosci. Remote Sens.*, vo1. 42, no. 10, pp. 2046–2057, Oct. 2004.
[26] S. Arisoy and K. Kayabol, "Mixture-based superpixel segmentation and classification of SAR images," *IEEE Geosci. Remote Sens.Lett.*, vo1. 13, no. 11, pp. 1721–1725, Nov. 2016.
[27] K. Kayabol and J. Zerubia, "Unsupervised amplitude and texture classification of SAR images with multinomial latent model," *IEEE Trans. Image Process.*, vol. 22, no. 2, pp. 561–572, Feb. 2013.
[28] Y. Li, M. Gong, L. Jiao, L. Li, and R. Stolkin, "Change-detection map learning using matching pursuit," *IEEE Trans. Geosci. Remote Sens.*, vo1. 53, no. 8, pp. 4712–4723, Aug. 2015.
[29] M Gong, J Zhao, J Liu, Q. Miao, et al, "Change detection in synthetic aperture radar images based on deep neural networks," *IEEE Trans. Neural. Netw. Learn. Syst.*, vol. 27, no. 1, pp. 125–138, January, 2016.
[30] B. Gu, V. S. Sheng, K. Y. Tay, W. Romano, et al, Incremental Support Vector Learning for Ordinal Regression, *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 7, pp. 1403–1416, 2015.
[31] B. Gu, X. M. Sun, and V. S. Sheng, Structural Minimax Probability Machine, *IEEE Transactions on Neural Networks and Learning Systems*, DOI : 10.1109/ TNNLS.2016. 2544779, 2016.
[32] B. Gu and V. S. Sheng, A Robust Regularization Path Algorithm for ν-Support Vector Classification, *IEEE Transactions on Neural Networks and Learning Systems*, DOI : 10.1109/TNNLS.2016.2527796, 2016.
[33] B. Chen, J. Yang, B. Jeon, X. Zhang, Kernel quaternion principal component analysis and its application in RGB-D object recognition, *Neurocomputing*, 10.1016/j.neucom.2017.05.047, 2017.
[34] J. Lu, D. Meng, S. Yu, Z. Lan, S. Shan, and A. G. Hauptmann, "Self-paced learning with diversity," in *NIPS*, 2014, pp. 2078–2086.
[35] M. P. Kumar, B. Packer and D. Koller, "Self-paced learning for latent variable models," in *NIPS*, 2010, pp. 1189–1197.
[36] M. Kumar, H. Turki, D. Preston, and D. Koller, "Learning specific-class segmentation from diverse data", in *ICCV*, 2001, pp. 1800–1807.
[37] J. Supancic III and D. Ramanan, "Self-paced learning for long-term tracking," in *CVPR*, 2013, pp. 2379–2386.
[38] Y. Lee and K. Grauman, "Learning the easy things first: self-paced visual category discovery," in *CVPR*, 2011, pp. 1721–1728.
[39] D. Meng and Q. Zhan. "What objective does self-paced learning indeed optimize? "*Compu. Sci.*, Nov. 2015.
[40] J. Liang, L. Jiang, D. Meng, and A. Hauptmann, "Learning to detect concepts from webly-labeled video data," in *IJCAI*, 2016.
[41] L. Jiang, D. Meng, T. Mitamura, and A. Hauptmann, "Easy samples first: self-paced reranking for zero-example multimedia search", in *ACMMM*, 2014.
[42] L. Jiang, D. Meng, Q. Zhao, S. Shan, and A. Hauptmann, "Self-paced curriculum learning," in *AAAI*, 2015.
[43] C. Li, F. Wei, J. Yan, W. Dong, Q. Liu, X. Zhang, and H. Zha, "A self-paced regularization framework for multi-label learning," in *arXiv*, Apr. 2016.
[44] C. Li, F. Wei, J. Yan, W. Dong, Q. Liu, and H. Zha, "Self-paced multi-task learning," AAAI, 2017.
[45] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *ICML*, 2009, pp. 41–48.
[46] S. Basu and J. Christensen, "Teaching classification boundaries to hunmans," in *AAAI*, 2013.
[47] J. Lu, D. Meng, S. Yu, Z. Lan, S. Shan, and A. G. Hauptmann, "Self-paced learning with diversity," in *NIPS*, 2014, pp. 2078-2086.
[48] M. Tan, I. W. Tsang, and L. Wang, "Minimax sparse logistic regression for very high–dimensional feature selection," *IEEE Trans. Neural. Netw. Learn. Syst.*, vol. 24, no. 10, pp. 1609–1622, Oct. 2013.
[49] S. Chakrabartty, R. K. Shaga, and K. Aono, "Noise-sharping gradient descent-based online adaptation algorithms for digital calibration of analog circuits," *IEEE Trans. Neural. Netw. Learn. Syst.*, vol. 24, no. 4, pp. 554–565, April. 2013.
[50] M. Gong, L. Su, M. Jia, and W. chen, "Fuzzy clustering with a modified MRF energy function for change detection in synthetic aperture radar images," *IEEE Trans. Fuzzy syst.*, vol. 22, no. 1, pp. 98-109, Feb. 2014.
[51] P. L. Rosin and E. Ioannidis, "Evaluation of global image thresholding for change detection," *Pattern Recog. Lett.*, vol. 24, no. 14, pp. 2345–2356, Oct. 2003.
[52] G. H. Rosenfield and K. Fitzpatrick-Lins, "A coefficient of agreement as a measure of thematic classification accuracy," *Photogram. Eng. Remote Sens.*, vol. 52, no. 2, pp. 223–227, 1986.

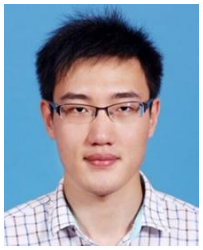


Ronghua Shang received the B.S. degree in information and computation science and the Ph.D. degree in pattern recognition and intelligent systems from Xidian University in 2003 and 2008, respectively.She is currently a professor with Xidian University. Her current research interests include optimization problems, evolutionary computation, image processing, and data mining.

Yijing Yuan received the B.E. degree in School of Computer Science and Control Engineering from North University of China, Taiyuan, China. Now she is pursuing the M.S. degree in School of Electronic Engineering from Xidian University, Xi'an, China. Her current research interests include image processing and machine learning.

Licheng Jiao (SM'89) received the B.S. degree from Shanghai Jiaotong University, Shanghai, China, in 1982, the M.S. and Ph.D. degrees from Xi'an Jiaotong University, Xi'an, China, in 1984 and 1990, respectively. From 1990 to 1991, he was a postdoctoral Fellow in the National Key Laboratory for Radar Signal Processing, Xidian University, Xi'an, China. Since 1992, Dr. Jiao has been a Professor in the School of Electronic Engineering at Xidian University. Currently, he is the Director of the Key Lab of Intelligent Perception and Image Understanding of Ministry of Education of China at Xidian University, Xi'an, China. Dr. Jiao is a Senior Member of IEEE, member of IEEE Xi'an Section Execution Committee and the Chairman of Awards and Recognition Committee, vice board chairperson of Chinese Association of Artificial Intelligence, councilor of Chinese Institute of Electronics, committee member of Chinese Committee of Neural Networks, and expert of Academic Degrees Committee of the State Council.

His research interests include image processing, natural computation, machine learning, and intelligent information processing. He has charged of about 40 important scientific research projects, and published more than 20 monographs and a hundred papers in international journals and conferences.



Yang Meng received the B.E. degree in electronic information engineering from Jiangsu University of Science and Technology, Zhenjiang, China. He is currently working toward master-doctor continuous study that the master's degree in electronic circuit and system and the doctor's degree in intelligent information processing from Xidian University, Xi'an, China. His current research interests include image processing and data mining.

Amir Masoud Ghalamzan received B.S. degree and M.S. degree in mechanical engineering in Iran. He received second level specialization degree in automatic control engineering from Politecnico di Torino and PhD in robotics from Politecnico di Milano in 2010 and 2015. From 2015 he is a postdoctoral fellow at the University of Birmingham. His research interests include robotic manipulation, robot learning from demonstration, machine learning and computer vision.